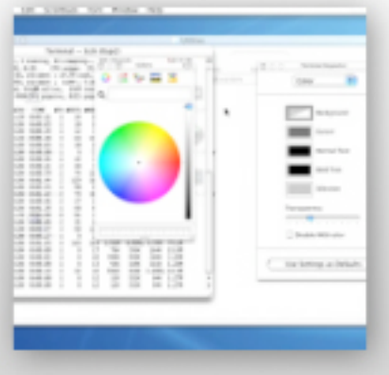
Security: Certificate APIs

Session 109



Security: Certificate APIs

Craig Keithley
Security Technology Evangelist
keithley@apple.com

Introduction

- Certificates and their uses
- Certificate management

What You'll Learn

- Certificates and identities
- Storing and retrieving certificates
- Establishing trust
- Displaying certificates and User Trust settings
- SecureTransport and User Trust

Security: Certificate APIs

Michael Brouwer
Sr. Software Engineer, OS Services

Technology Framework

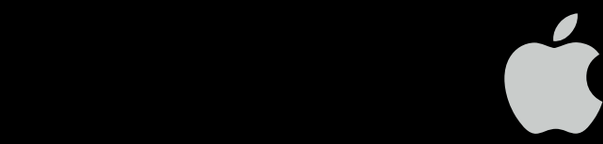
- Security.framework
  - SecKeychain
  - SecCertificate
  - SecIdentity
  - SecTrust
- New in Panther
  - SecurityFoundation
  - SecurityInterface
- SecurityHICocoa is gone

What You Will Learn

- Overview of CDSA
- Basic functionality provided by Security framework
- Using certificates to establish trust
- Using keychains to store certificates

CDSA Overview

- Modular plug-in architecture for security services
  - Cryptographic Service Provider (CSP)
  - Datastore Library (DL)
  - Certificate Library (CL)
  - Trust Policy (TP)
- Common Security Services Manager (CSSM) provides API access



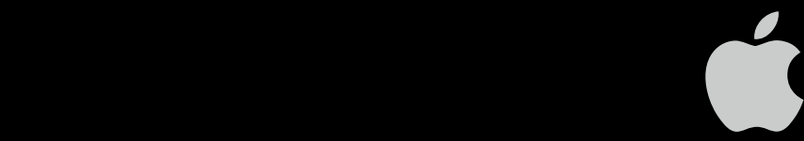
CDSA Architecture

CDSA Functionality

- Key generation (symmetric and asymmetric)
- Encryption (AES, etc.)
- Digests (MD5, SHA-1)
- Data storage and retrieval
- Access control on keys
- Certificate parsing and evaluation

Security Framework

- Implements CDSA functionality in Mac OS X
- Provides CDSA interfaces
- Provides higher-level Security interfaces
- User interface factored out for low-level access
  - UI is done in SecurityAgent
- Part of Darwin (open source)



About “Sec” APIs

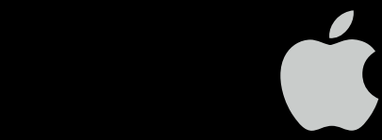
- Patterned after CoreFoundation
- In fact, they are CF objects
- Can use CFRetain, CFRelease, CFArray, etc.
- Be aware of Copy vs. Get
- Bridge functions to CSSM

Sec APIs

File Signing APIs
Keychain APIs
Layered Services
Authorizing APIs
Certificate APIs
SecureTransport (SSL)

Keychain APIs

- Keychain Manager
- SecKeychain
- Both APIs are supported
- SecKeychainRef = KCRef (since 10.2)
- May bring up UI (via SecurityAgent) unless user interaction is off



SecKeychain Advantages

- Supports all application environments
- More flexible; all attributes of a keychain item can be specified at once
- Access control support
- System Keychain support

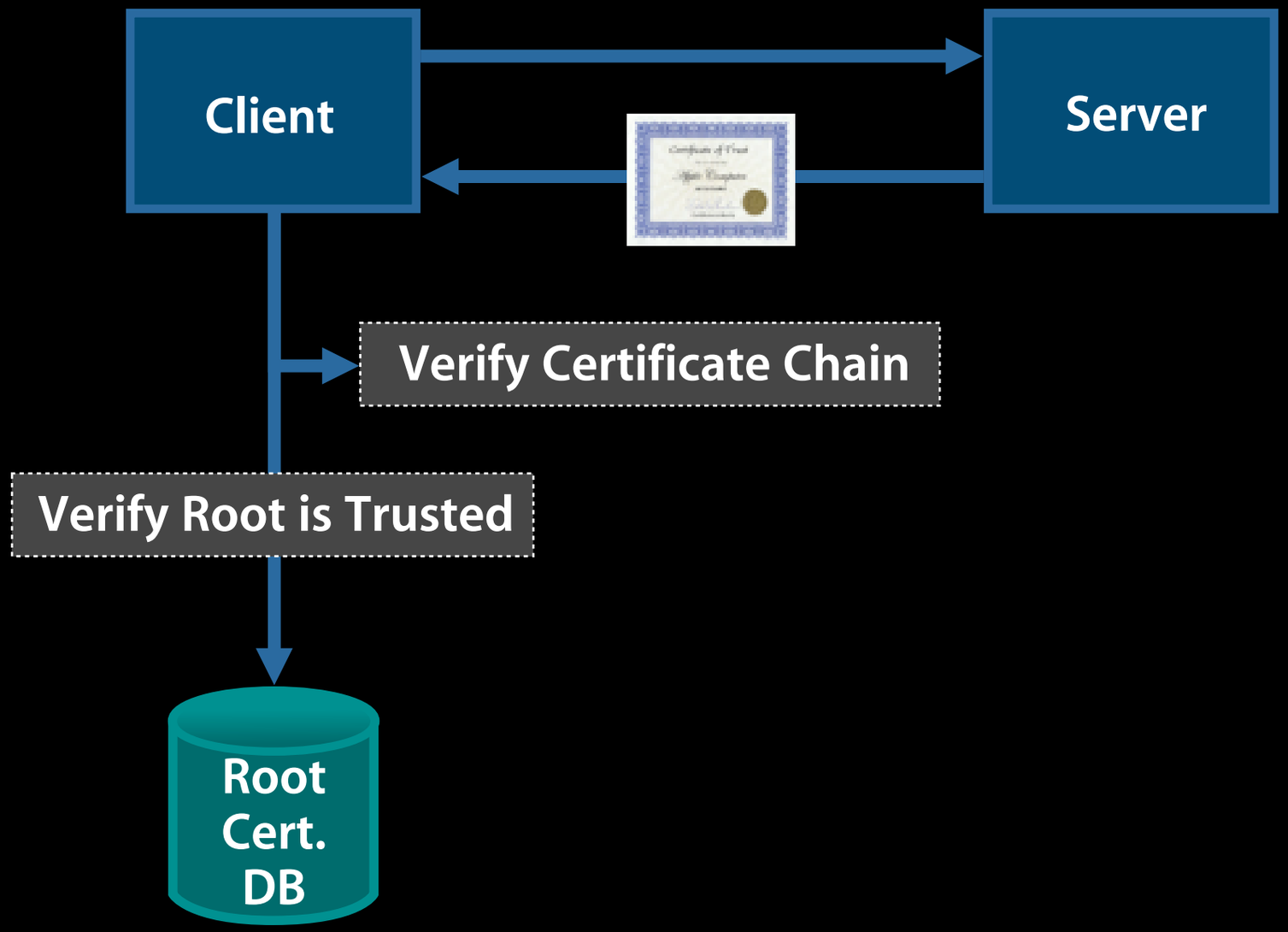
What's in a X.509 Certificate?

- Public Key
- Attributes (for example, issuer, subject, validity)
- Extensions
  - Critical (for instance, key usage, policies)
  - Non critical
- Signature of all of the above

X.509 Certificates

- Certificates are signed by their issuers
  - The contents of the certificate can't be tampered with unless you have access to the signing key
- Intermediate certificates sign other certificates
- Root certificates sign themselves
  - Root certificates trusted by prior arrangement
  - Root certificate database

Example: SSL Usage



SecCertificate

- "Sec" wrapper for a certificate
- SecCertificateRef operations
  - Import and export raw data
  - Store in and retrieve from Keychains
  - Use a CL module to parse it
  - Display using SFCertificateView
  - Determine trustworthiness for a particular operation (Using SecTrust)

SecCertificate (Cont.)

- Used in APIs
  - SecureTransport (SSL)
  - Verification (SecTrust)
  - Identities (SecIdentity)
- No access control (do not panic!)

SecIdentity

- Represents a user identity
- Consists of Private Key and a Certificate
  - Stored separately in keychains
- Access control through private key
- Good advice
  - Let user choose
  - Rely on system for access control

Uses of SecIdentity

- SecureTransport client side authentication
- 802.1x (VPN)
- Future applications coming

Policies

- Rules for determining trust
- Examples of policies
  - SSL Policy
  - S/MIME signing
  - Code signing
- Some policies take parameters

SecTrust

- Purpose
  - Validate certificate
  - Establish user trust for an policy
  - Do this as simply as possible
- SecTrust is a workflow object
  - Create it
  - Add your ingredients
  - Ask it to evaluate

SecTrust (Cont.)

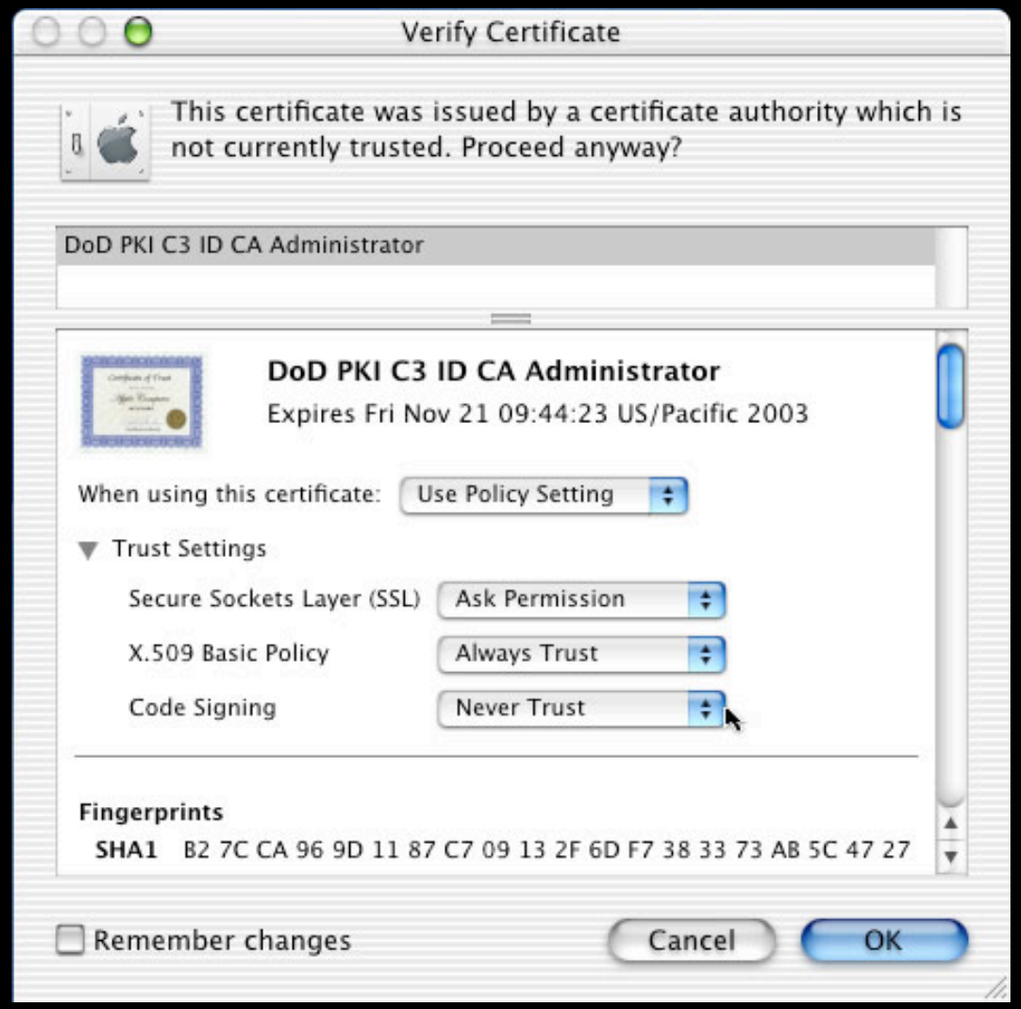
- Outcomes
  - Valid user choice (yes, no, ask, do not know)
  - Recoverable and fatal errors
  - More results on the side
- Ingredients
  - Policies and policy parameters
  - Provide helpful certificates
  - Search keychains for more certificates
  - Anchor certificates

SecUserTrust

- Management API for user trust settings
- Only useful for administrative applications
- UI layer allows user to manage trust directly
- Trust is stored separately by:
  - Certificate
  - Operation
- Trust Values are Yes, No, Ask, and Unset

Editing Certificate Trust

- UI available from Carbon and Cocoa
- Lets users make trust decisions for specific policies

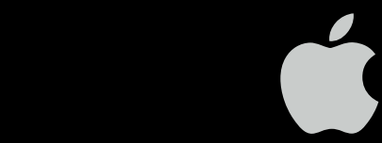


Demo

Ken McLeod
Sr. Software Engineer, OS Services

Summary

- New and improved APIs for your pleasure
  - Available in Jaguar
  - For Mac OS X 10.1, stick to Carbon Keychain Manager API
- Easier than CSSM layer (but less flexible)
  - Experts can “bridge down” to CDSA
  - Pick what you need, ignore the rest
  - Avoid micro-management



Wrap Up

Roadmap

101 Security Architecture Infrastructure The Security Infrastructure in Mac OS X	**The Marina** **Mon., 5:00pm**
108 Kerberos in Mac OS X	**The Mission** **Thu., 9:00am**
FF016 Security Feedback Forum Tell Us What You Think	**North Beach** **Fri., 2:00pm**

Who to Contact

Craig Keithley
Security Technology Evangelist
keithley@apple.com

http://developer.apple.com/wwdc2003/urls.html

For More Information

Reference Library

Documentation

Certificate, Key, and Trust Services Reference

ADC Home > Documentation > Security >

Other Resources

ADC Security page

http://developer.apple.com/security/index.html

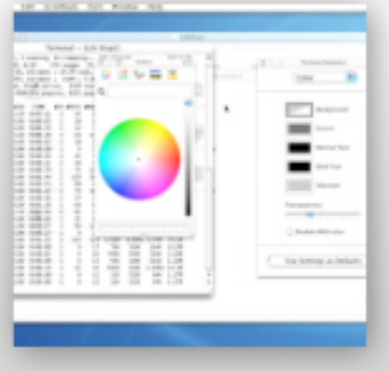
Apple CDSA mailing list

http://lists.apple.com/mailman/listinfo/apple-cdsa

Common Data Security Architecture

http://www.opensource.apple.com
http://www.opengroup.org

Q&A



Craig Keithley
Security Technology Evangelist
keithley@apple.com

http://developer.apple.com/wwdc2003/urls.html

WWDC 2003

Worldwide Developers Conference